TD8 : projet Partie 1 - conception

Objectifs du TD : apprendre à anticiper et conceptualiser un projet complet en programmation

Chargés de TD :

- Anne-Julie Tinet
- Alexis Derycke

(alexis.derycke@univ-lorraine.fr)

Déroulé du TD :

- intro + rappel (30-45 min)
- conceptualisation (45min-1h)
- architecture (30-45min)

Objectif : mettre en application tous les concepts dans le cadre d'un projet

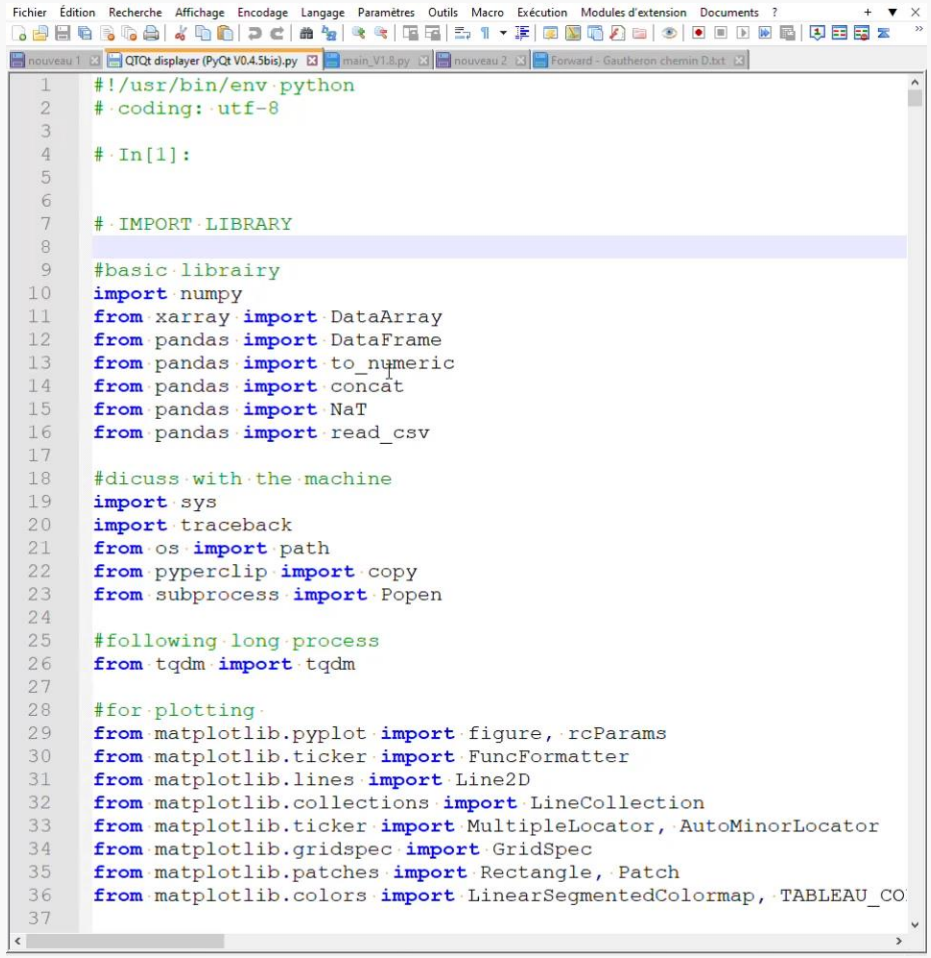
Évaluation :

- groupe de 2
- rendu d'un programme + un document informatif (cf. détails)
- travail en TD (2 séances) + personnel

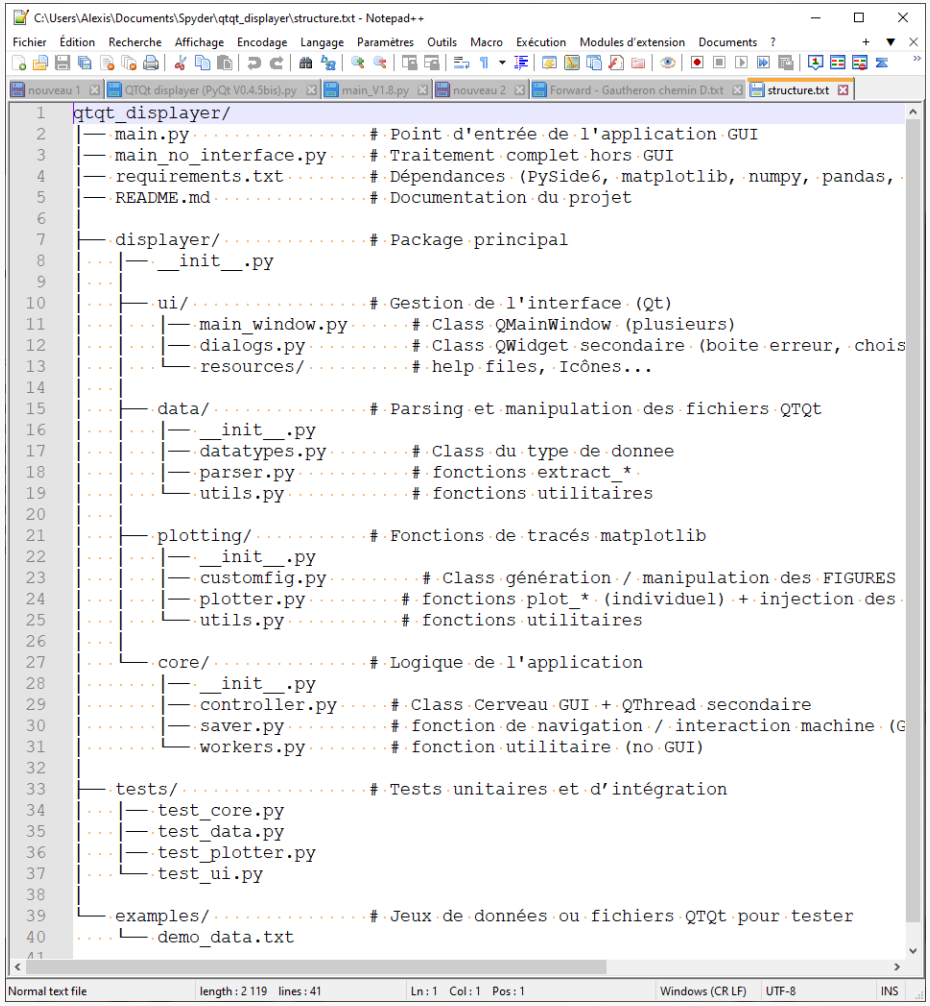
Déroulé :

- 1ère séance : aide à la conception
- 2e séance : aide à la programmation

Rappel S5-TD1 : l'algorithmie, concrètement c'est quoi ?



```python
#!/usr/bin/env python
# coding: utf-8

# In[1]:


# IMPORT LIBRARY

#basic librairy
import numpy
from xarray import DataArray
from pandas import DataFrame
from pandas import to_numeric
from pandas import concat
from pandas import NaT
from pandas import read_csv

#dicuss with the machine
import sys
import traceback
from os import path
from pyperclip import copy
from subprocess import Popen

#following long process
from tqdm import tqdm

#for plotting
from matplotlib.pyplot import figure, rcParams
from matplotlib.ticker import FuncFormatter
from matplotlib.lines import Line2D
from matplotlib.collections import LineCollection
from matplotlib.ticker import MultipleLocator, AutoMinorLocator
from matplotlib.gridspec import GridSpec
from matplotlib.patches import Rectangle, Patch
from matplotlib.colors import LinearSegmentedColormap, TABLEAU_CO
```



```
qtqt_displayer/
├── main.py                    # Point d'entrée de l'application GUI
├── main_no_interface.py       # Traitement complet hors GUI
├── requirements.txt           # Dépendances (PySide6, matplotlib, numpy, pandas,
├── README.md                  # Documentation du projet
│
├── displayer/                 # Package principal
│   ├── __init__.py
│   │
│   ├── ui/                    # Gestion de l'interface (Qt)
│   │   ├── main_window.py     # Class QMainWindow (plusieurs)
│   │   ├── dialogs.py         # Class QWidget secondaire (boite erreur, chois
│   │   └── resources/         # help files, Icônes...
│   │
│   ├── data/                  # Parsing et manipulation des fichiers QTQt
│   │   ├── __init__.py
│   │   ├── datatypes.py       # Class du type de donnee
│   │   ├── parser.py          # fonctions extract_*
│   │   └── utils.py           # fonctions utilitaires
│   │
│   ├── plotting/              # Fonctions de tracés matplotlib
│   │   ├── __init__.py
│   │   ├── customfig.py       # Class génération / manipulation des FIGURES
│   │   ├── plotter.py         # fonctions plot_* (individuel) + injection des
│   │   └── utils.py           # fonctions utilitaires
│   │
│   └── core/                  # Logique de l'application
│       ├── __init__.py
│       ├── controller.py      # Class Cerveau GUI + QThread secondaire
│       ├── saver.py           # fonction de navigation / interaction machine (G
│       └── workers.py         # fonction utilitaire (no GUI)
│
├── tests/                     # Tests unitaires et d'intégration
│   ├── test_core.py
│   ├── test_data.py
│   ├── test_plotter.py
│   └── test_ui.py
│
└── examples/                  # Jeux de données ou fichiers QTQt pour tester
    └── demo_data.txt
```

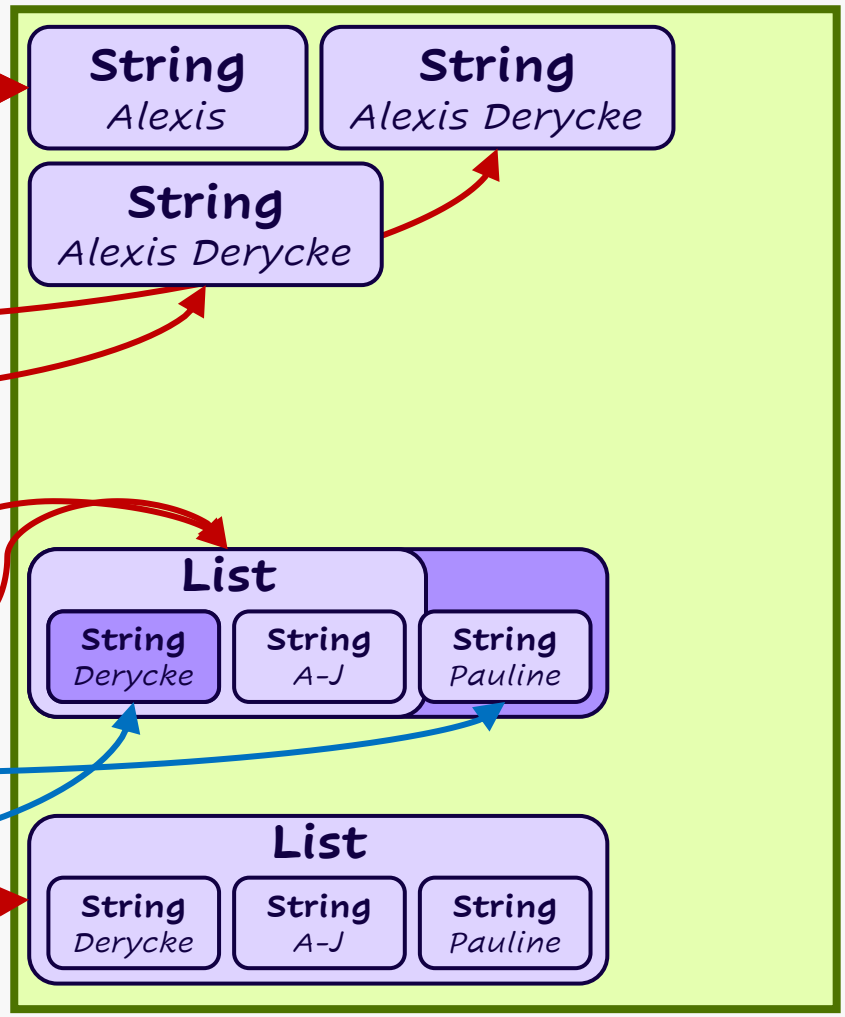
Rappel S5-TD2 : Python : la gestion des variables

IMMUTABLE (IMMUABLE)

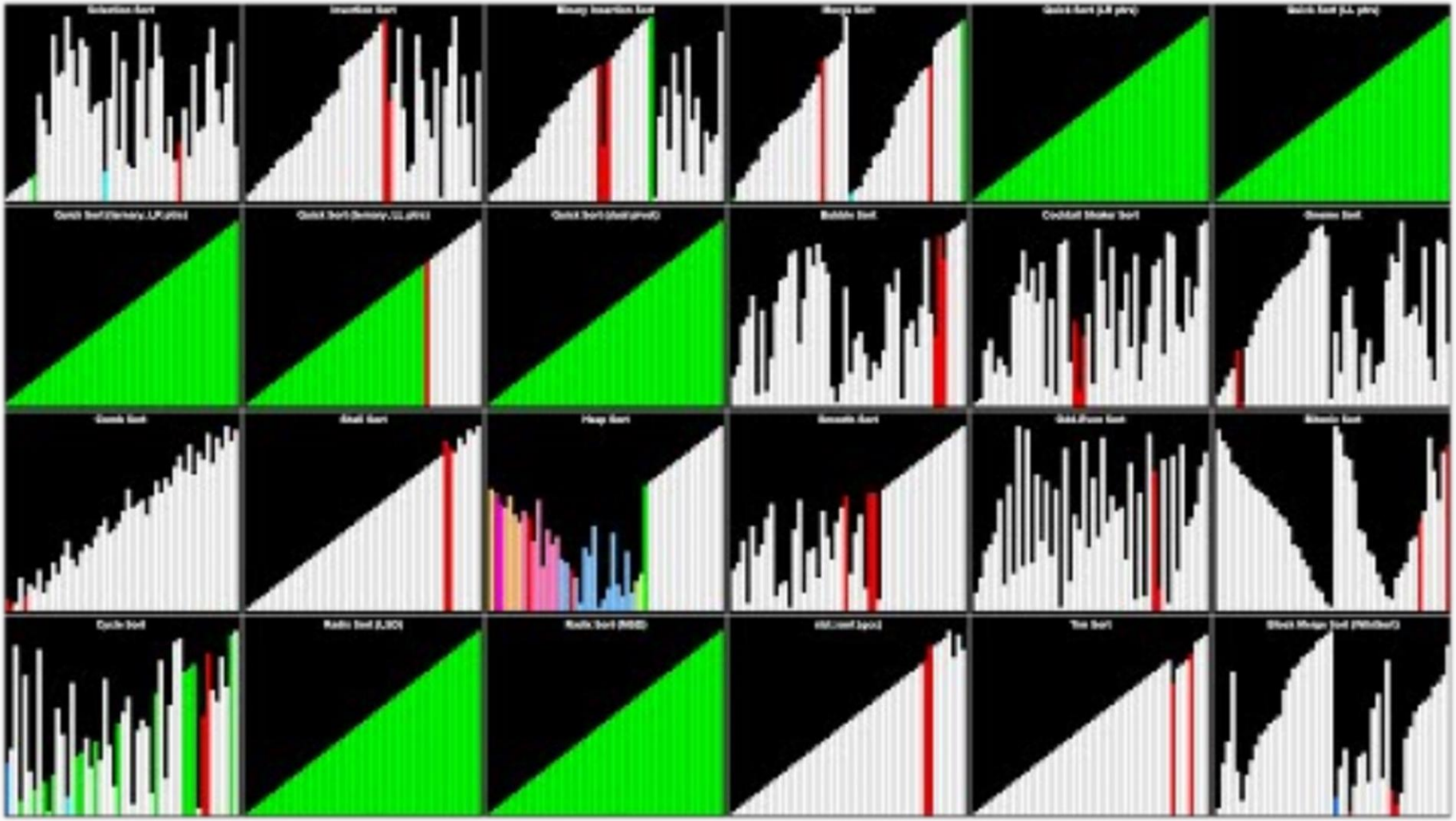
```
Prof = ”Alexis”
Prof_bis = Prof
Prof_ter = Prof + “ Derycke”
Prof_bis = “Alexis Derycke”
```

MUTABLE (MUABLE)

```
List_Prof = [”Alexis”, “Anne-Julie”]
List_Prof_bis = List_Prof
List_Prof_bis.append(“Pauline”)
List_Prof_bis[0] = “Derycke”
List_Prof_ter = List_Prof.copy()
```



Rappel exemples d'algorithme
exercice du tri : LE problème historique



Visualization of 24 Sorting Algorithms In 2 Minutes - Viktor Bohush

Rappel S6-TD2 : comment en arrive-t-on à la POO ?

Processus de programmation :

1) Idée & prototypage

2) Squelette & V0.1

3) Amélioration

4) Production (V1) & développement

papier & test python

1er fichier python & librairy

Projet python, fonctions, multi fichier ...

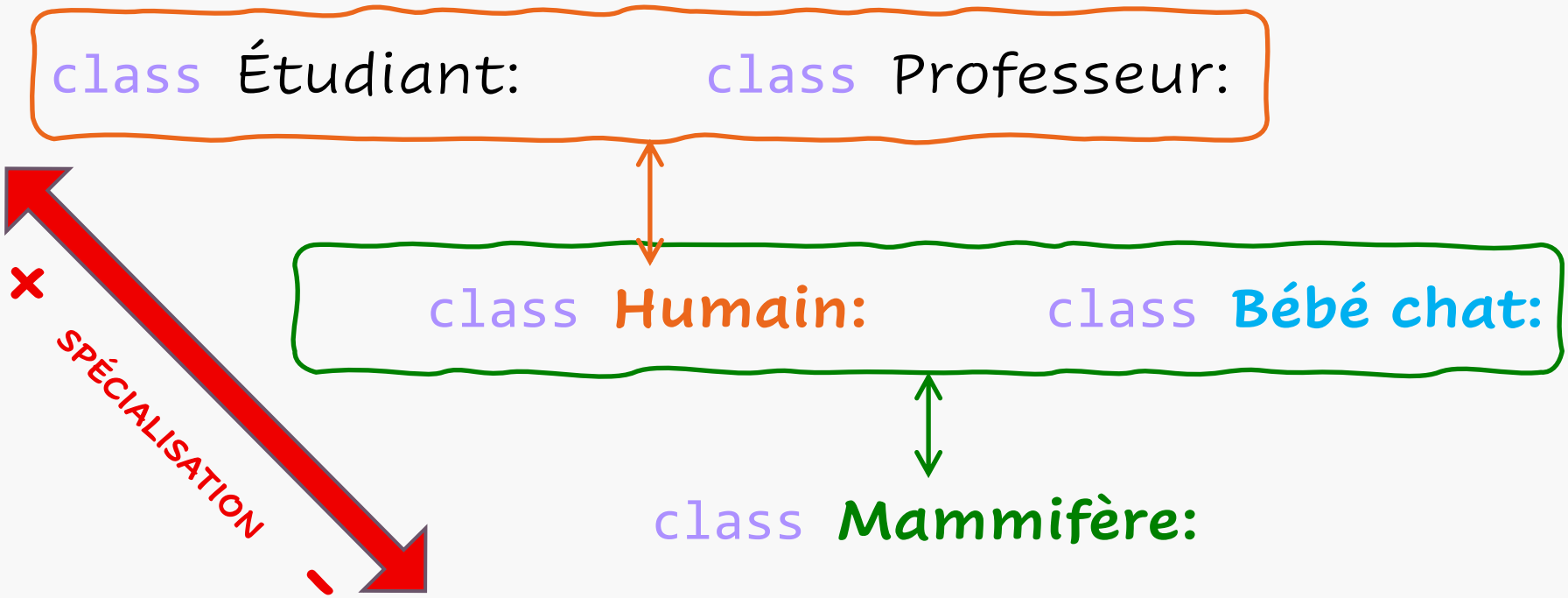
Création de class (POO)

Rappel S6-TD3 : l'héritage - qu'est-ce que c'est ?

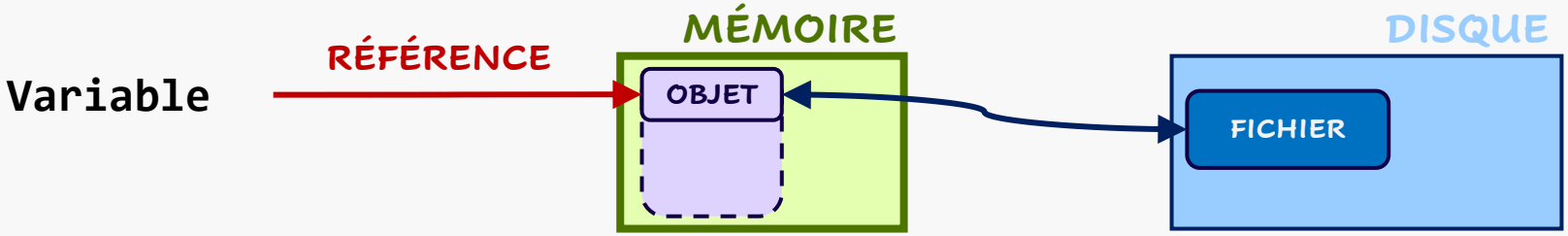
Attention :

programmation ≈≠ filiation

Principe :



Rappel S6-TD5 : gestion des fichiers



Manipulation suivant les 3 actions :

ouverture > lecture/écriture > fermeture

« mise en mémoire »

« accès/modification du fichier au travers de la mémoire »

« libération de la mémoire »

-> attention à bien fermer les fichiers pour libérer l'espace mémoire !

Résumer des connaissances

À mettre en pratique :

- Algorithme efficace
- Bonne gestion de la mémoire
- Bonne gestion des fichiers
- Programmation suivant les bonnes pratiques

} S5 et S6

À découvrir :

- Conception de la structure d'un programme complexe
- Conception de la POO complexe

L'éléphant dans la pièce....

L’éléphant dans la pièce…. l’IA

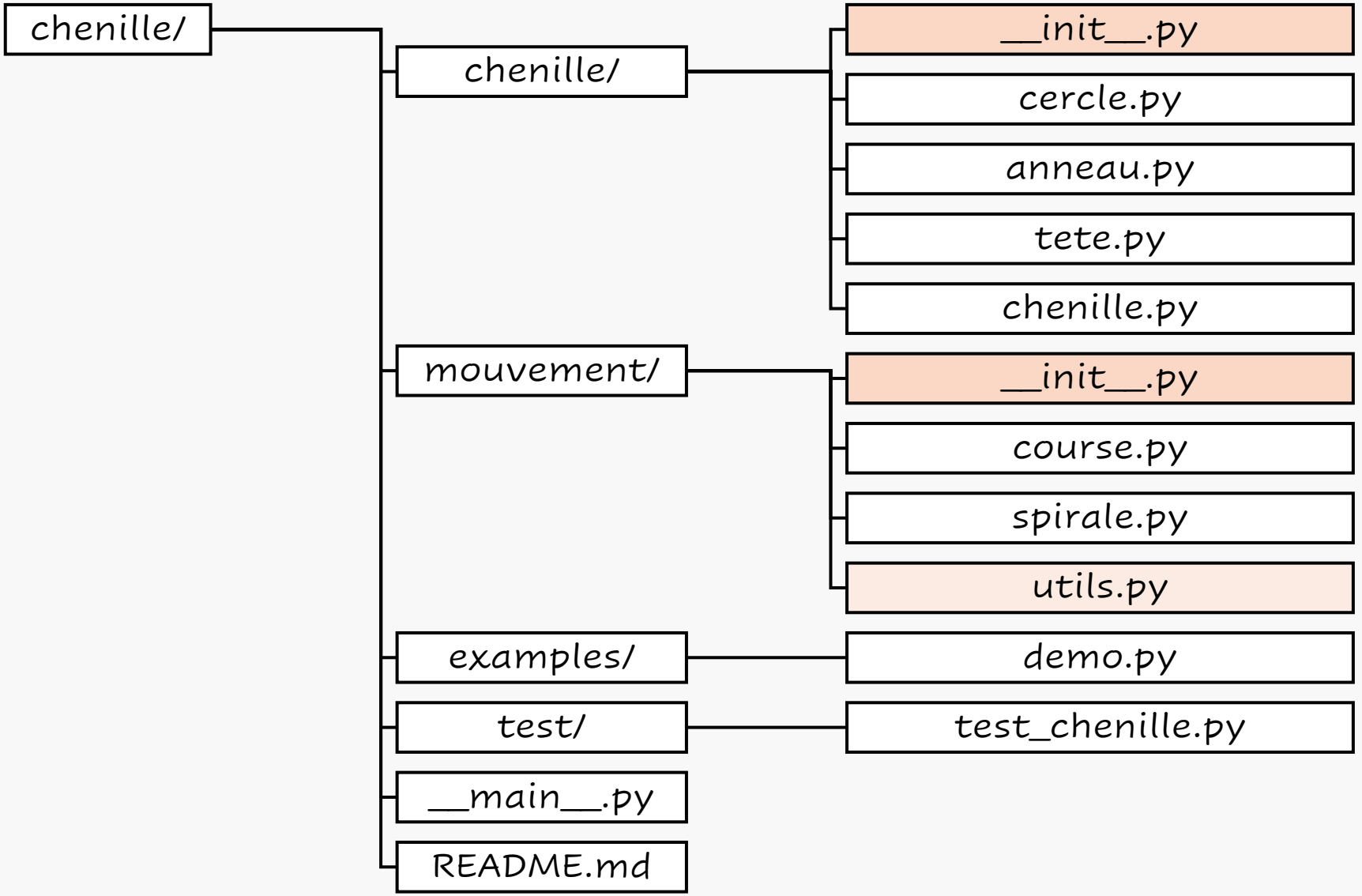
L’éléphant dans la pièce…. l’IA

L'éléphant dans la pièce.... l'IA

Démonstration avec la chenille et chatGPT (free) :

saurais-tu me faire un code python qui affiche en chenille avec turtle et qui la déplace jusqu'au bord de l'écran ?

L'éléphant dans la pièce .... l'IA pas si intelligent que ça...



L'éléphant dans la pièce.... l'IA comment l'utiliser ?

Niveau 0	Niveau 1	Niveau 2	Niveau 3	Niveau 4
Utilisation **interdite**	Utilisation **limitée**	Utilisation **guidée**	Utilisation **balisée**	Utilisation **libre**

INTERDIT

ASSISTANCE

AMÉLIORATION

PRODUCTION

RÉALISATION

Le rendu

1) le programme développé, sous forme de scripts Python (.py) enregistrés dans un zip

2) Un document annexe indiquant :

- Une auto-évaluation sur les bonnes pratiques
- La gestion de la programmation par le binôme
  - ✓ répartition des tâches
  - ✓ temps de travail respectif
  - ✓ contrôle mutuel des éléments de programmation
- Dans le cas de l'utilisation d'assistance (de tout type : enseignant, autre étudiant, personne extérieure, AI)
  - ✓ à quelles questions à répondu cette assistance
  - ✓ comment cela s'intègre dans votre travail
  - ✓ comment avez-vous contrôlé ces éléments

L'évalutation

Un bon programme est :

Fiable

Simple

Performant

AF1

Lisible

Maintenable

Collaboratif

AF2

L'évaluation

- ✓ Mon programme répond aux objectifs de l'énoncé
- ✓ Mon programme applique les méthodologies exigées dans l'énoncé
- ✓ Chaque élément (fonction, méthode, etc.) a été validé, et ces validations sont vérifiables par un utilisateur extérieur

- ❖ Je souhaite apporter des améliorations / fonctionnalités supplémentaire : OK, mais les fonctionnalités exigées doivent être aussi présentes
- ❖ J'ai identifié un problème de fiabilité, mais je n'arrive pas à le résoudre : préciser le problème et les solutions qui ont été tentées dans le document annexe

L'évaluation

✓Mon programme **respecte les conventions** de la PEP-8 (nommage, espacements, etc.)

✓Mon programme est **structuré de façon modulaire**, selon les pratiques vues dans les séances de TD précédentes

✓Mon programme **est documenté** et commenté de façon adéquate